Resilience of Deployed TCP to Blind Attacks

Luckie, M. *et al.*,
Proc. of ACM IMC '15, pp. 13-26, 2015.

Allison McDonald
Xinghao Li

Introduction

- TCP is one of the most widely used transport layer protocol.
- However, it was built vulnerable to attacks (RFC 793).
- There are some defences for blind in-window attacks (RFC 5961)
- Modern TCP protocol stack is still vulnerable
  - Web servers
  - Infrastructure

Contributions of this paper

- Reveals the vulnerability of TCP connection
- Measures the vulnerability of TCP connection in real network.
- Introduces possible defences for TCP in-window attack

Outline

- TCP Background
- Measurement method
- Web Server vulnerability
- Infrastructure vulnerability
- Port selection observations
- Conclusion
- Discussion

Background - TCP

- 4-Tuple
  - Source IP address/Port number
  - Destination IP address/Port number
- SEQ
  - Must be in-window to be accepted
- ACK
- Flags
  - SYN
  - RST
  - FIN

Background - TCP Connection Establishment

- 3-Way Handshake

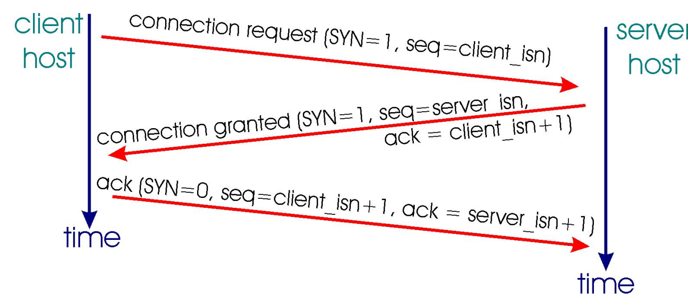


Figure 1[2]

Background - TCP Connection Termination

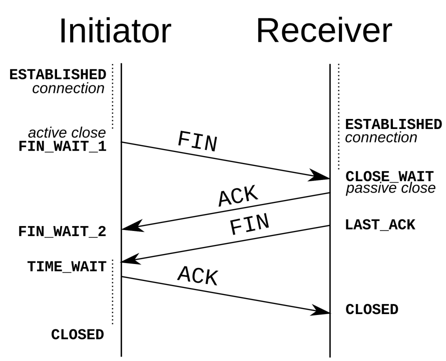


Figure 2[1]

Background - TCP Connection Reset

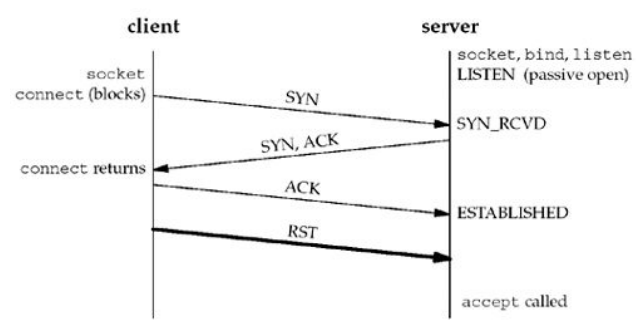


Figure 3[3]

TCP Blind In-window Attacks

- Reset
- SYN
- Data Injection

TCP Blind In-window Attack

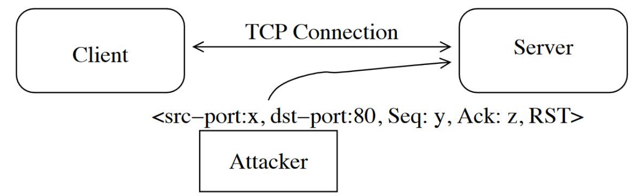
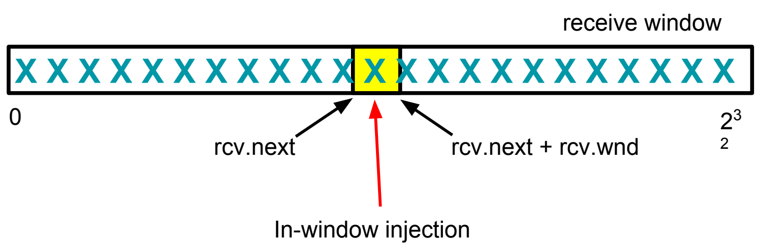


Figure 4[4]

Slipping in the Window

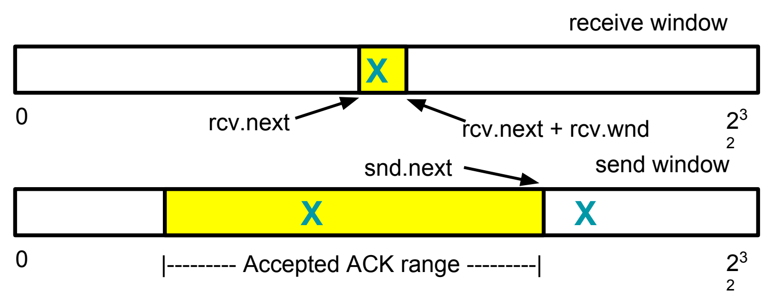
"a reset is valid if its sequence number is in the window" - RFC 793



Based on slides by Luckie, IMC'15

Slipping in the Window

"an acknowledgement value is acceptable as long as it is not acknowledging data that has not yet been sent" - RFC 793



Based on slides by Luckie, IMC'15

Defenses

- Making port number hard to guess
  - Using random ephemeral port numbers
- Require the sequence number be more accurate
  - RFC 5961
- Filtering the spoofed IP address at origin (RFC 2827)
- For BGP
  - Generalized TTL Security Mechanism (GTSM)
  - TCP MD5

RFC 5961 vs RFC 793 - Reset

If (RST.seq in accepted range):

    Reset connection

Else:

    Send Ack packet

RFC 793

If (RST.seq = expected seq):

    Reset connection

Else if (RST.seq in accepted range):

    Send Challenge Packet

Else:

    Drop the packet, do nothing

RFC 5961

RFC 5961 vs RFC 793 - SYN

If (SYN.seq in accepted range):

    Reset connection

Else:

    Send Ack packet

RFC 793

Send challenge packet

If (received a RST packet):

    Reset connection

Else:

    Drop the packet, do nothing

RFC 5961

RFC 5961 vs RFC 793 - Data Injection

- For a data packet to be accepted:

- SEQ in accepted range
- ACK in accepted range
  - (SND.UNA-2^31+1, SND.NXT)

RFC 793

If (SEQ not in accepted range):

    Drop the packet

If (ACK in (**SND.UNA-SND.MAX.WIN, SND.NXT**)):

    Accept the packet

Else if (ACK in (**SND.UNA-2^31+1,SND.UNA-SND.MAX.WIN**):

    Send challenge packet

Else: Drop the packet

RFC 5961

RFC 5961 - Accepted ACK Range

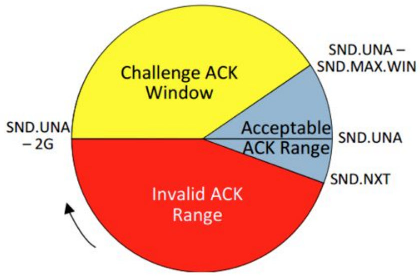
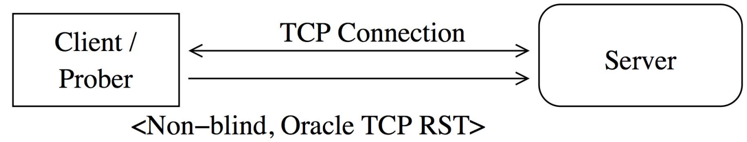


Figure 6[6]

Experimental Setup



Measurement Method - RST and SYN

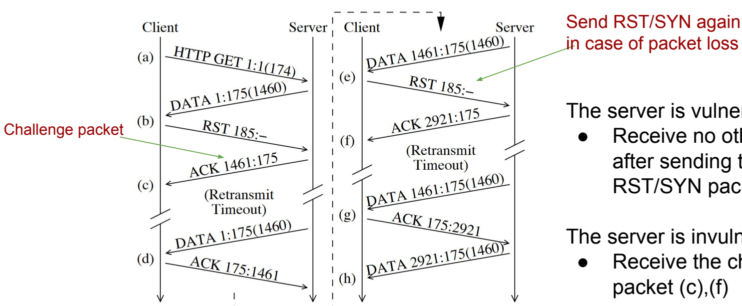


Figure 7[4]

The server is vulnerable if

- Receive no other packet after sending the first RST/SYN packet(b)

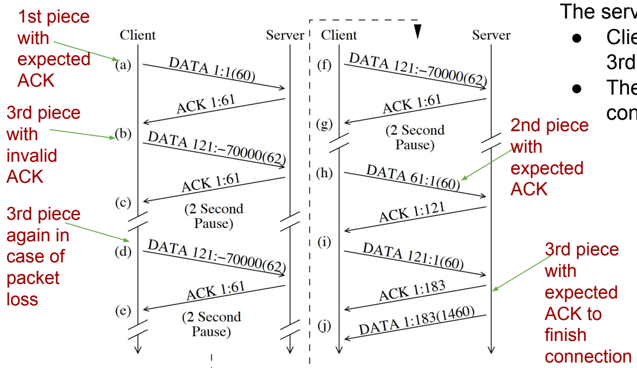
The server is invulnerable if

- Receive the challenge packet (c),(f)

Measurement Method - Data

- Idea: Divide the first segment of data into three pieces
  - Some servers (22%) reset the connection if receiving unexpected ACK number **for the first segment of data**, without checking the SEQ number.
  - They do not send a reset packet for subsequent data packets with unexpected ACK number.

Measurement Method - Data



The server is vulnerable if

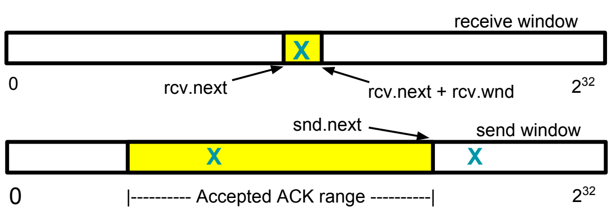
- Client receive ACK for the 3rd piece (after h)
- The server reset the connection

Figure 8[4]

Testing Web Server Vulnerability

- Target
  - Alexa Top 1,000,000
- Vantage Point
  - CAIDA's Archipelago in US and New Zealand
  - Machine at MIT

What was tested?



Methods	Parameters
Blind reset: in window	Host's snd.nxt + 10
Blind reset: out of window	Host's snd.nxt - 70,000
Blind SYN: in window	Host's snd.nxt + 10
Blind SYN: out of window	Host's snd.nxt - 70,000
Blind data: behind	Peer's snd.una - 70,000
	Host's snd.nxt + x
Blind data: ahead	Peer's snd.una + 70,000
	Host's snd.nxt + x

Results

Result	Blind reset		Blind SYN		Blind data	
	in	out	in	out	behind	ahead
Accepted	3.4%	0.4%	-	-	29.6%	5.4%
Reset (ack-blind)	-	-	17.1%	0.0%	0.6%	0.6%
Reset (dup-ack)	18.8%	0.6%	5.3%	1.2%	0.1%	0.2%
Vulnerable	22.2%	1.0%	22.4%	1.2%	30.3%	6.2%
Challenge ACK	71.4%	1.1%	37.7%	57.0%	37.1%	8.1%
Ignored	5.1%	91.8%	35.9%	38.3%	29.3%	81.3%
Not Vulnerable	76.5%	93.0%	73.6%	95.3%	66.4%	89.4%
Parallel TCP	-	-	1.1%	1.1%	-	-
Early FIN	0.3%	3.3%	1.5%	1.6%	3.2%	3.7%
No Result	1.0%	2.7%	1.3%	0.9%	0.1%	0.7%
Other	1.3%	6.0%	4.0%	3.6%	3.3%	4.4%

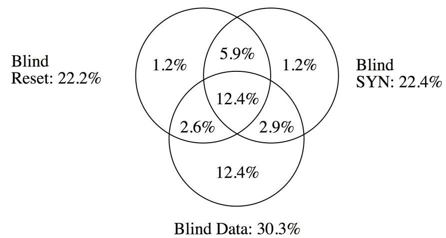
Results from US vantage point

Results

	cld-us	MIT	hlz-nz
Blind reset (in):			
Vulnerable	22.2%	22.1%	21.9%
Not Vulnerable	76.5%	76.0%	76.5%
Other	1.3%	1.9%	1.6%
Blind SYN (in):			
Vulnerable	22.4%	22.2%	0.3%
Not Vulnerable	73.6%	73.2%	94.2%
Other	4.0%	4.6%	5.5%
Blind data (behind):			
Vulnerable	30.3%	30.3%	30.3%
Not Vulnerable	66.4%	66.5%	66.2%
Other	3.3%	3.3%	4.5%

Summary of results from all vantage points

38.4% vulnerable to at least one attack!



Overlap of vulnerabilities

Results

Operating System	Blind reset in	Blind reset out	Blind SYN in	Blind SYN out	Blind data behind	Blind data ahead	Total
FreeBSD 8.x	19.2%	0.5%	**93.8%**	56.5%	**83.9%**	None	193 (0.5%)
FreeBSD 9.x	18.8%	1.0%	**88.1%**	22.2%	54.7%	None	612 (1.5%)
Linux 2.4-2.6	**87.4%**	3.0%	**83.6%**	0.4%	54.3%	40.5%	269 (0.6%)
Linux 2.6.x	**90.1%**	0.9%	**84.1%**	None	63.2%	35.8%	4903 (11.8%)
Linux 3.x	15.3%	0.6%	14.0%	0.1%	11.6%	0.6%	18021 (43.4%)
Windows 7 or 8	5.1%	2.1%	0.3%	0.3%	**88.7%**	0.9%	3877 (9.3%)
Windows XP	7.9%	6.1%	3.0%	1.8%	6.3%	3.5%	838 (2.0%)
Unknown	9.6%	0.8%	12.7%	1.4%	23.9%	3.2%	12543 (30.2%)

Vulnerability to blind attacks by operating system

Middleboxes Defenders?

Server MSS	Vulnerable Portion: Blind reset	Blind SYN	Blind data
1460 (87.2%)	23.9%	24.7%	28.1%
1380 (5.4%)	2.0%	0.5%	58.8%
8961 (2.3%)	2.3%	2.3%	4.7%
1440 (0.8%)	5.9%	4.7%	57.5%
1436 (0.7%)	22.2%	5.8%	32.5%

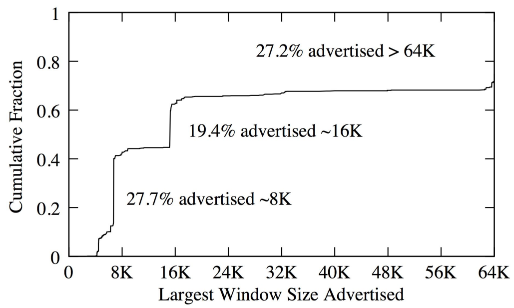
Maximum Segment Size and vulnerability

Middleboxes Defenders?

Server MSS	Vulnerable Portion: Blind reset	Blind SYN	Blind data
1460 (87.2%)	23.9%	24.7%	28.1%
1380 (5.4%)	2.0%	0.5%	58.8%
8961 (2.3%)	2.3%	2.3%	4.7%
1440 (0.8%)	5.9%	4.7%	57.5%
1436 (0.7%)	22.2%	5.8%	32.5%

Maximum Segment Size and vulnerability

Window Sizes



Largest window size for servers vulnerable to in-window attacks

Infrastructure Vulnerability

- BGP and OpenFlow both have long-lived TCP connections
  - More time for attacker to probe the connection!
  - Disruption could be harmful
- Some mitigating measures
  - Generalized TTL Mechanism (GTSM)
  - TCP cryptographic authentication
  - Traffic filtering from untrusted networks
- Testing in the wild not possible (or advisable)

Infrastructure Vulnerability

Device	OS	Blind reset		Blind SYN		Blind data		Port
	date	in	out	in	out	behind	ahead	range
Cisco 2610 12.1(13)	2002-01	× (A)	✓ (I)	× (R)	✓ (C)	× (A)	✓ (C)	seq.
Cisco 2610 12.2(7)	2002-01	× (A)	✓ (I)	× (R)	✓ (C)	× (A)	✓ (C)	seq.
Cisco 2650 12.3(15b)	2005-08	✓ (C)	✓ (I)	✓ (C)	✓ (C)	× (A)	✓ (C)	40785
Cisco 7206 12.4(20)	2008-07	✓ (C)	✓ (I)	✓ (C)	✓ (C)	× (A)	✓ (C)	54167
Cisco 2811 15.0(1)	2010-10	✓ (C)	✓ (I)	✓ (C)	✓ (C)	× (A)	✓ (C)	46166
Cisco 2911 15.1(4)	2012-03	✓ (C)	✓ (I)	✓ (C)	✓ (C)	× (A)	✓ (C)	39422
Juniper M7i 8.2R1.7	2007-01	× (A)	✓ (I)	× (R)	✓ (I)	× (A)	✓ (C)	181
Juniper EX9208 14.1R1.10	2014-06	✓ (C)	✓ (I)	✓ (C)	✓ (I)	× (A)	✓ (C)	13769
Juniper MX960 13.3	2015-05	✓ (I)	✓ (I)	✓ (C)	✓ (I)	× (A)	✓ (C)	13033
Juniper J2350 12.1X46-D35.1	2015-05	✓ (I)	✓ (I)	✓ (C)	✓ (I)	× (A)	✓ (C)	12481
HP 2920 WB.15.16.0006	2015-01	✓ (C)	✓ (C)	✓ (C)	✓ (C)	✓ (I)	✓ (I)	14273
HP e3500 K.15.16.0007	2015-06	× (A)	✓ (I)	× (R)	✓ (C)	✓ (I)	✓ (I)	15611
Brocade MLX-4 5.7.0bT177	2014-10	✓ (I)	✓ (I)	✓ (C)	✓ (C)	✓ (C)	✓ (C)	const.
Pica8 Pronto3290 v2.6	2015-05	× (A)	✓ (I)	× (R)	✓ (C)	× (A)	× (A)	HBPS

A: accepted
R: reset
C: challenged
I: ignored

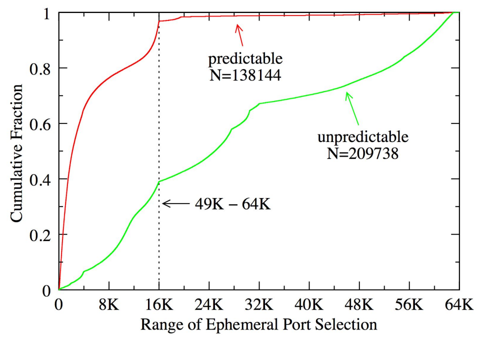
Laboratory tests of TCP attacks against BGP-speaking
routers and OpenFlow-speaking switches

Ephemeral Port Selection

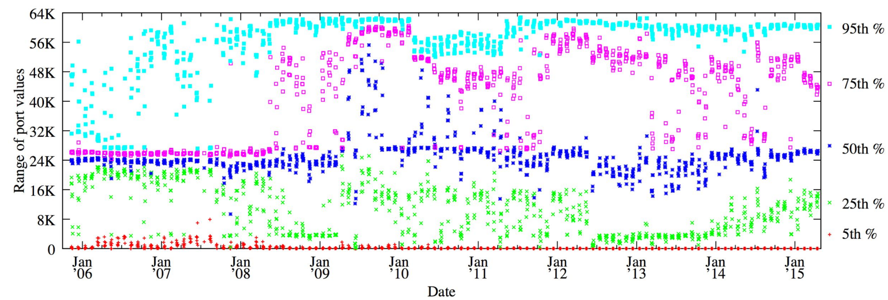
- How predictable are ephemeral ports?
- Packet traces at a network tap!
  - Find source IPs with >10 connections and that transferred data
  - With a sliding window of 3, determine whether ports generally increasing
    - Increasing: [1,2,3], [2,3,1], [3,1,2];
    - Not: [2,1,3], [3,2,1], [1,3,2]
  - If all windows increasing, classify as predictable!

Ephemeral Port Selection

- Range of ports
  - (max - min)
- 50% stay in a range of 2K!

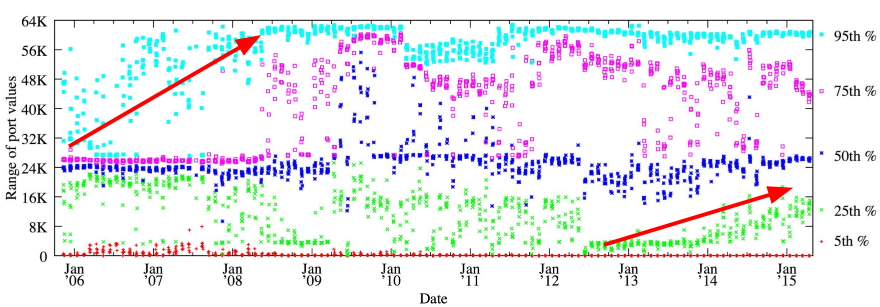


Ephemeral Port Selection Range



Range of ports for one day of Bro logs collected one week per month at ICSI

Ephemeral Port Selection Range



Range of ports for one day of Bro logs collected one week per month at ICSI

Improvements

- Another defence for TCP blind in-window attacks?
  - Random port number selection
  - RFC 5961
    - Is it safe?
- How vulnerable are client OSes?
  - MacOS was < 0.5% of tested servers; not included in study

Discussion

- Why do some OSes not follow RFC 5961?
- Why is there variation in vulnerability in the same OS?

References

[1]https://en.wikipedia.org/wiki/Transmission_Control_Protocol

[2]http://www2.ic.uff.br/~michael/kr1999/3-transport/3_05-segment.html

[3]http://www.masterraghu.com/subjects/np/introduction/unix_network_programming_v1.3/ch05lev1sec11.html

[4]Luckie, M. *et al.*, "Resilience of Deployed TCP to Blind Attacks," *Proc. of ACM IMC '15*, pp. 13-26, 2015.

[5]http://www.hackingaccount.com/what-is-tcp-syn-flood-attack/?EsetProtoscanCtx=2313f10c980

[6]http://www.myhack58.com/Article/html/3/62/2016/78614.htm